

Contextual Security Analysis (CSA)



Firstly, a big THANK YOU! to:

- Troy Marshall
- Sean Poris
- Ben Pick
- Mike McCabe
- Rinaldi Rampen

- ALL OF YOU! 🙌🙌🙌

Ken Johnson @cktricky

Co-Host of Absolute AppSec
CTO of DryRun Security
BJJ Nerd



Common language / terminology

Dynamic Application Security Testing (DAST)	Sending requests to running web applications and then observing & analyzing the application's behavior.
Static Application Security Testing (SAST)	Analysis of the source code, in its a non-execution environment, for patterns & anti-patterns that indicate a potential security issue.
Contextual Security Analysis (CSA)	A modern, comprehensive risk assessment of software changes using a multitude of factors/data-points.
AI / LLM	Witch magic 🧙‍♂️

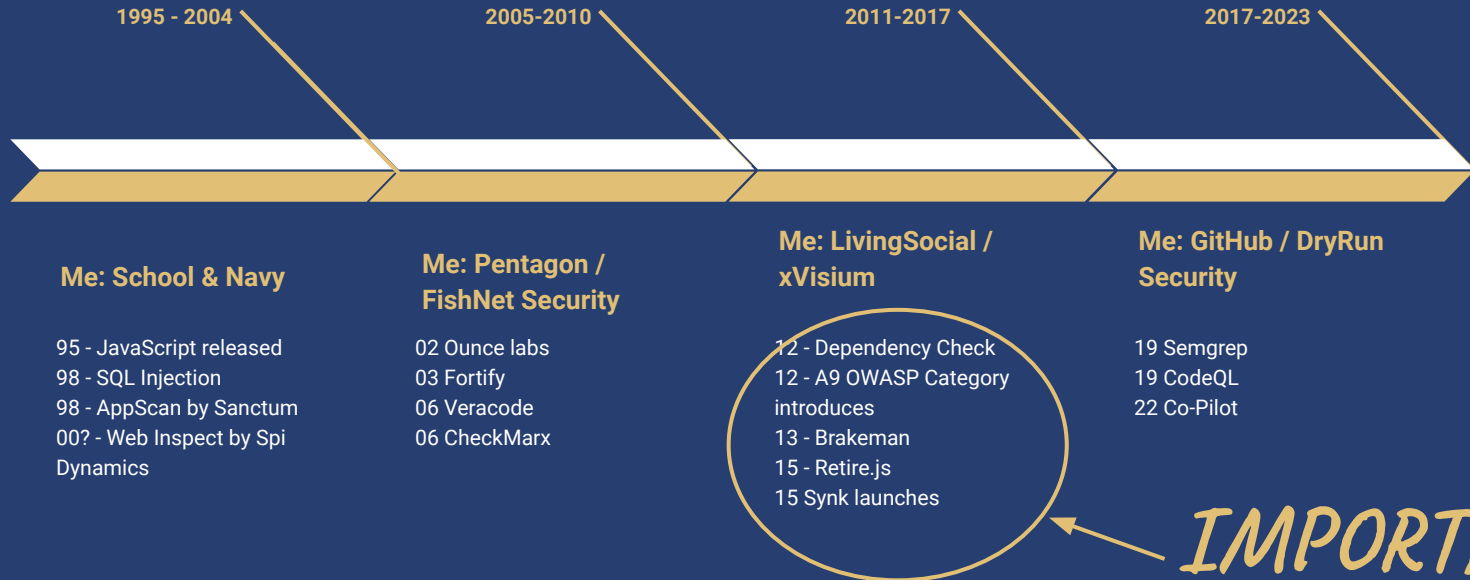
Outline

- A Brief History: DAST, SAST & Me
- Introducing Contextual Security Analysis
 - Current examples
 - Overview of CSA
- S.L.I.D.E.
 - Surface
 - Language/Framework
 - Intent
 - Detection
 - Environment
- LLMs / AI
 - Use cases
 - Examples

A Brief History:

SAST/DAST
&
Me

A Brief History: SAST, DAST, & Me



Takeaways

💻 DAST sort of dropped off along the way...

👤 Tools marketed with developers in mind

⚙️ Tools still built with Security expertise required (problematic for Security Champions)

🙋 **Perfection in DAST/SAST does not exist.**
Our tolerance for mistakes is way higher
than we'd like to admit

Known Gaps

Technology stacks are adopted faster than we can write tools

Scanners of today are a singular, mixed-result accuracy data point

Completely ignores the following data points:

- Previously discovered vulnerabilities
- Known dangerous portions of a codebase
- Code Quality risk markers
- Authorship completely ignored as a risk factor
- Existing Service Risk
- ... The list goes on (S.L.I.D.E.)

Conclusion



Write something:

- ✓ Practical for people defending
- ✓ Harnesses existing tooling & pipeline
- ✓ Build on top of existing tribal knowledge
- ✓ Balance between adaptability/speed and perfection

Contextual Security Analysis (CSA)

Tell me you're doing
CSA without telling me
you're...

Practical Use Cases

Awareness:

Looking for new risks in code changes using more than one factor

Variant Analysis & Regressions:

Looking for similar issues that have cropped up in the past

Remediation:

Assist developers

Chime & Monocle

SECURITY SCORE FACTORS	
Docker image vulnerabilities goal: zero criticals/highs	0% of 5%
Gem security issues goal: zero criticals/highs	✓
Approved base images	✓
Brakeman enabled	✓
Bundler-audit enabled	✓
Default branch protected	✓
Github security alerts goal: zero criticals/highs	✓
Security Scan enabled	✓
CircleCI enabled	✓
CodeClimate enabled	✓
CodeClimate security issues goal: zero criticals/highs	✓
Updated within 1 month	✓
To score full marks, please make frequent updates. This reduces the number of vulnerabilities in dependencies and in application code.	



<https://medium.com/life-at-chime/monocle-how-chime-creates-a-proactive-security-engineering-culture-part-1-dedd3846127f>



USES MULTIPLE RISK FACTORS ✨



Practical for people defending



Harnesses existing tooling & pipeline



Balance between adaptability/speed and perfection

<https://github.com/cktricky/csa/tree/OWASP-NoVA>

GitHub Advanced Security (GHAS)

<https://docs.github.com/en/code-security/getting-started/github-security-features#available-with-github-advanced-security>

REVEALS MULTIPLE RISK FACTORS:

- Dependency scanning
- CodeQL results
- Secret Scanning

Available with GitHub Advanced Security

The following GitHub Advanced Security features are available and free of charge for public repositories on GitHub.com. Organizations that use GitHub Enterprise Cloud with a license for GitHub Advanced Security can use the full set of features in any of their repositories. For a list of the features available with GitHub Enterprise Cloud, see the [GitHub Enterprise Cloud documentation](#).

Code scanning

Automatically detect security vulnerabilities and coding errors in new or modified code. Potential problems are highlighted, with detailed information, allowing you to fix the code before it's merged into your default branch. For more information, see "[About code scanning](#)."

Secret scanning alerts for users

Automatically detect tokens or credentials that have been checked into a repository. You can view alerts for any secrets that GitHub finds in your code, in the **Security** tab of the repository, so that you know which tokens or credentials to treat as compromised. For more information, see "[About secret scanning](#)."

Dependency review

Show the full impact of changes to dependencies and see details of any vulnerable versions before you merge a pull request. For more information, see "[About dependency review](#)."



<https://github.com/cktricky/csa/tree/OWASP-NoVA>

Practical Use Cases

Awareness:

Looking for new risks in code changes using more than one factor

Variant Analysis & Regressions:

Looking for similar issues that have cropped up in the past

Remediation:

Assist developers

Semgrep



 <https://www.iit.io/blog/semgrep-to-uncover-log4j-vulnerabilities>

- ✓ Practical for people defending
- ✓ Prioritizes speed (important when an incident occurs)
- ✓ Harnesses existing tooling & pipeline



```
root@ /home/lahiru/SemgrepTest# semgrep --config=auto /home/lahiru/code/

Scan Status

Scanning 15 files tracked by git with 1065 Code rules:

Language Rules Files Origin Rules
<multilang> 51 30 Community 1065
Java 116 2
yaml 27 1
dockerfile 4 1
bash 4 1

100% 0:00:00

1 Code Finding

/home/lahiru/code/log4shell-vulnerable-app-main/README.md
generic.secrets.security.detected-aws-access-key-id-value.detected-aws-access-key-id-value
AWS Access Key ID Value detected. This is a sensitive credential and should not be hardcoded
here. Instead, read this value from an environment variable or keep it in a separate,
private file.
Details: https://sg.run/Ged1

22| ...
ACTIAIWDYXACS5H53AN2F20211211W3Fus-east-1X2Fz3X2Faus4_request&X-Amz-Date=20211211T031401Z&X-Amz-Expires
=300&X-Amz-Signature=140e57e1827c6f42275aa ...
[shortened a long line from output, adjust with --max-chars-per-line]

Scan Summary

Ran 1065 rules on 15 files: 1 finding.
root@ /home/lahiru/SemgrepTest#
```

<https://github.com/cktricky/csa/tree/OWASP-NoVA>

Practical Use Cases

Awareness:

Looking for new risks in code changes using more than one factor

Variant Analysis & Regressions:

Looking for similar issues that have cropped up in the past

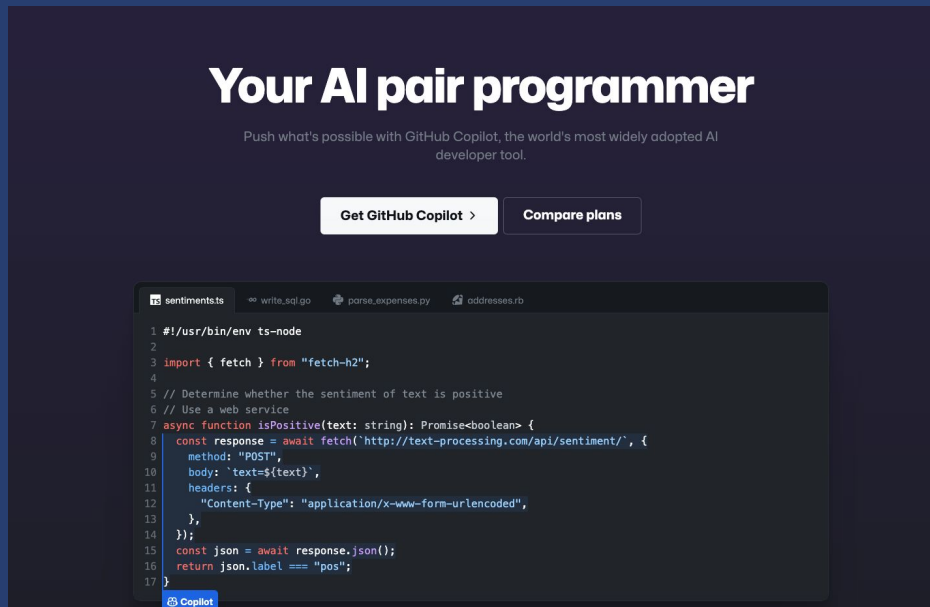
Remediation:

Assist developers

Co-Pilot

 <https://github.com/features/copilot>

- ✓ Practical for people defending
- ✓ Prioritizes speed over perfection
- ✓ Harnesses existing tooling & pipeline




Semgrep Assistant

 <https://semgrep.dev/products/semgrep-code/assistant/>

```
252 +         logger.error(  
253 +             f"Found rule with no id in registry. Rule is:\n '{str(item)}'",  
254 +         )
```

jbergler marked this conversation as resolved. ⚙ Hide resolved

 semgrep-app (bot) 16 hours ago • edited ▾

The first parameter when logging with structlog is the event name.
Follow these rules when writing an event name:

- use only lowercase letters, underscores, and dots, because these events get indexed in Datadog, and spaces or special characters make searching and browsing in a spreadsheet view more difficult
- use a maximum of 50 characters
- do not do string formatting, instead, pass variables as keyword arguments for nicer indexing in Datadog

Semgrep Assistant thinks this is a **true positive** that should be fixed. If you agree, a fix for this finding might look like this:

Suggested change

```
252 -         logger.error(  
253 -             f"Found rule with no id in registry. Rule is:\n  
254 -             '{str(item)}'",  
252 +         logger.error(  
253 +             "rule_without_id_found",  
254 +             rule=str(item),  
255 +             message="Found rule with no id in registry.",  
256 +         )  
257 +         continue
```

Commit suggestion ▾

Key concepts 💡

Ideally, we merge these strategies together

There are still signals for risk that we're not yet tapping into

We need tooling to keep pace with internal technology adoption

Scaling knowledge is key

Overview

Context > Control

Control

(S,D,I)AST embodies this:

- 😭 Limited data points
- 😭 Enforcement of rules
- 😭 Blocking checkpoints
- 😭 Pattern matching

Context

Next Gen:

- 😍 Combines many data points
- 😍 Warnings & guidance over enforcement
- 😍 Remediation assistance
- 😍 Risk-focused

Download Link: <https://www.dryrun.security/resources/csa-guide>

CSA - SLIDE ☐

Surface	How the surface of the application changes
Language	Language and framework that the application is written in
Intent	Evaluates the person making the change, both in their patterns and their purpose
Detection	Tooling in place to detect vulnerabilities and security issues
Environment	Purpose of the application and service in the organization

SLIDE - Surface

Sensitive Code Paths

Depending on your language and framework, this could be configuration files, middleware, controllers, or how your application and service handles auth.

HTTP Routes



Routes refers to an exposed endpoint on an application that will take user-supplied information and perform an action. Expansion or contractions of these routes in an application inherently change the risk profile of the application.

SLIDE - Language

Source Language

This weighs the language used (ex: Golang, Rust, PHP, etc.) as each has their own security issues and known vulnerabilities.

Web Framework

Each web framework contains their own unique variations of sensitive areas, configurations, and components. Framework nuances are subtle and highly specific security issues that are akin to “footguns”  .

SLIDE - Intent



Authorship frequency

If the author rarely commits or has never committed to this repo this indicates the author may be unfamiliar with the development standards of the code base.

□ Author is not an owner

Code changes to a portion of code in which the author is not an owner or on a team that owns this portion of the code.

SLIDE - Detection

Secrets

Secret scanning technology has enabled the detection of sensitive keys when pushed in source code. This is a major indicator of risk in a code change.

Testing

Consuming (S|D)AST, dependency, docker image, etc. scanner tool output. Past bug bounty submissions as a historical reference to known dangerous anti-patterns. Similar concept with vulnerabilities Identified by internal/external security contributors.

SLIDE - Environment

Change protection

Does the application utilize branch protection features or other testing-relevant gating mechanisms to ensure evaluation of new code changes?

Business Risk

Critical factors include the level of risk a service poses through its business function, the type of data it processes/stores/transmits, and its overall importance to the business.

!(Deterministic vs Probabilistic)

Surprise... it's both! 🎉

Probabilistic	<p>Use to gather context (ie - is this touching authentication?)</p> <p>Use to provide (anti)pattern analysis 🧠 (ie - does this match the pattern for a previously known 🐛)</p>
Deterministic	<p>Also can be used to gather context (ie - Write code that checks if this is a first time repo author)</p> <p>Specific things that produce exact matches (ie - Write code / Semgrep pattern)</p>
🏆 USE THE OUTPUT OF BOTH TO REVEAL RISK 🏆	

AI & You

Personal recommendations



Start experimentation with Baseplate or something similar



Read this article:

<https://engineering.peerislands.io/extending-openai-gpt-4-using-langchain-and-pinecone-for-q-a-over-your-own-content-using-1f3e9dc10e91>



Read this book:

<https://www.amazon.com/What-ChatGPT-Doing-Does-Work/dp/1579550819>

Langchain

Modular Components

- LLMs
- Compressors
- Retrievers
- Prompts
- Chains
- ... and way more



My AI learnings...

Contextual Compression + Documents, Chains, LLMs, Prompts... they are all highly configurable and can be very accurate

Created the following:

- AI Chat bot assistant that can dig through your organization's secure code docs to answer questions
- Analysis agent that provides composition of the application (framework, language, security centric libraries, datastore, etc.)
- Analysis agent that looks for known (in your app's specific code base) security bug patterns

Baseplate - Create Dataset

Dataset:

Juice Shop's README.md and package.json files

Datasets > Compositional Analysis Saved ✓ 9d314e6f-edbb-438d-8e5d-75de106f61c0 Save			
Compositional Analysis Double click a row to edit your embeddings. Documents Query Standard Search			
Description (optional)			
+ rowid	text (embedded) Search...	documentid Search...	url Search...
9142372	# [Juice Shop Logo] (https://raw.githubusercontent.com/juice-shop/juice-shop/master/frontend/src/assets/public/images/juice-shop_100px.png) OWASP Juice Shop [[OWASP Flagship] (https://img.shields.io/badge/owasp-flagship%20project-48A46.svg)] (https://owasp.org/projects/#sec-	25da97cf-b794-4b98-8468-962f511b9c90	https://service.baseplate.ai/storage/v1/lobb-96b3-4700-b62b-da32128dfab/readme.md?token=eyJ2GGoOUlUzNlNlNlR5CCGkxkYVC-9g4Ad7b2VTGoeV3vXt82KvY58
9142373	[CICD Pipeline] (https://github.com/juice-shop/juice-shop/workflows/CICD%20Pipeline/badge.svg?branch=master) [[Test Coverage] (https://api.codeclimate.com/v1/badges/6206c8f3972bcc97033f/test_coverage) (https://codeclimate.com/github/juice-shop/juice-shop/test_coverage)] [[Maintainability]	25da97cf-b794-4b98-8468-962f511b9c90	https://service.baseplate.ai/storage/v1/lobb-96b3-4700-b62b-da32128dfab/readme.md?token=eyJ2GGoOUlUzNlNlNlR5CCGkxkYVC-9g4Ad7b2VTGoeV3vXt82KvY58
9142374	[GitHub star] (https://img.shields.io/github/stars/juice-shop/juice-shop.svg?label=GitHub%20EZ%20%20style=flat) [[Contributor Covenant] (https://img.shields.io/badge/Contributor%20Covenant-v2.0%20adopted-f66e4.svg)] (CODE_OF_CONDUCT.md)	25da97cf-b794-4b98-8468-962f511b9c90	https://service.baseplate.ai/storage/v1/lobb-96b3-4700-b62b-da32128dfab/readme.md?token=eyJ2GGoOUlUzNlNlNlR5CCGkxkYVC-9g4Ad7b2VTGoeV3vXt82KvY58
9142375	> [The most trustworthy online shop out there.] (https://twitter.com/ehackpurple/status/708781693504589824) > [[@ehackpurple] (https://github.com/ehackpurple)] > [The best juice shop on the whole internet!] (https://twitter.com/ehackpurple/status/90733535775085668) > [[@ehackpurple] (https://twitter.com/ehackpurple)]	25da97cf-b794-4b98-8468-962f511b9c90	https://service.baseplate.ai/storage/v1/lobb-96b3-4700-b62b-da32128dfab/readme.md?token=eyJ2GGoOUlUzNlNlNlR5CCGkxkYVC-9g4Ad7b2VTGoeV3vXt82KvY58
9142376	OWASP Juice Shop is probably the most modern and sophisticated insecure web application! It can be used in security trainings, awareness demos, CTFs and as a guinea pig for security tools! Juice Shop encompasses vulnerabilities from the entire [OWASP Top Ten] (https://owasp.org/www-project-top-ten/) along with	25da97cf-b794-4b98-8468-962f511b9c90	https://service.baseplate.ai/storage/v1/lobb-96b3-4700-b62b-da32128dfab/readme.md?token=eyJ2GGoOUlUzNlNlNlR5CCGkxkYVC-9g4Ad7b2VTGoeV3vXt82KvY58
9142377	## Table of contents - [Setup] ([#setup]) - [From Sources] ([#from-sources]) - [Packaged Distributions] ([#packaged-distributions]) - [Docker Container] ([#docker-container])	25da97cf-b794-4b98-8468-962f511b9c90	https://service.baseplate.ai/storage/v1/lobb-96b3-4700-b62b-da32128dfab/readme.md?token=eyJ2GGoOUlUzNlNlNlR5CCGkxkYVC-9g4Ad7b2VTGoeV3vXt82KvY58
9142378	### From Sources [GitHub repo size] (https://img.shields.io/github/repo-size/juice-shop/juice-shop.svg)	25da97cf-b794-4b98-8468-962f511b9c90	https://service.baseplate.ai/storage/v1/lobb-96b3-4700-b62b-da32128dfab/readme.md?token=eyJ2GGoOUlUzNlNlNlR5CCGkxkYVC-9g4Ad7b2VTGoeV3vXt82KvY58

Baseplate - Create App

Provide the following information:

Template

Context (point to a dataset)

Input Variables (Optional)

Start asking questions...

The screenshot displays the 'Create App' interface for a 'Chat App'. The interface is divided into several panels:

- Apps > Chat App**: Shows the app name and a 'Saved' status with a checkmark.
- Model**: A dropdown menu set to 'gpt-4'.
- Temperature**: A slider set to 0.7.
- Max tokens**: A slider set to 256.
- Top p**: A slider set to 1.
- Frequency penalty**: A slider set to 0.
- Presence penalty**: A slider set to 0.
- Rest of**: A slider set to 1.
- Variables**: A section for defining variables.
 - Context**: A dropdown menu set to 'content'.
 - Dataset**: A dropdown menu set to 'Demo'.
 - Query variables**: A section for selecting variables to use as the search query below. A variable named 'package' is selected.
 - Top K**: A dropdown menu set to 3.
 - Number of rows returned**: A slider.
 - Advanced options**: A link to expand more options.
 - Input**: A dropdown menu set to 'package'.
 - Use latest message**: A toggle switch.
- Template**: A section for defining the app's template.
 - Place variable names inside double curly braces {{curly braces}}. Variable**: A text input field.
 - Variables**: A section for defining variables. Two variables are defined: 'content' and 'package'.
 - You have been provided a package file to analyze. Provide**: A text input field.
 - use, and its overall composition.**: A text input field.
 - Answer the following question: {{content}}**: A text input field.
- Filled template**: A section for displaying the filled template. It shows a JSON object with 'role' and 'content' fields.

Baseplate - Ask questions

Filled template

Tokens: 941 Chat

Clear

```
[
{
  "role": "system",
  "content": "You have been provided a package file to analyze. Provide answers to the questions you are given about the application, its libraries in use, and its overall composition.\n\nAnswer the following question: {\\"finale-rest\\": \"^1.1.1\\\", \"fs-extra\\": \"^9.0.1\\\", \"fuzzball\\\": \"^1.3.0\\\", \"glob\\\": \"^7.1.6\\\", \"graceful-fs\\\": \"^4.2.6\\\", \"grunt\\\": \"^1.2.1\\\", \"grunt-contrib-compress\\\": \"^1.6.0\\\", \"grunt-replace-json\\\": \"^0.1.0\\\", \"hashids\\\": \"^2.2.1\\\", \"hbs\\\": \"^4.0.4\\\", \"helmet\\\": \"^4.0.0\\\", \"html-entities\\\": \"^1.3.1\\\", \"i18n\\\": \"^0.11.1\\\", \"js-yaml\\\": \"^3.14.0\\\", \"jsonwebtoken\\\": \"^0.4.0\\\", \"jssha\\\": \"^3.1.1\\\", \"juicy-chat-bot\\\": \"^0.8.0\\\", \"libxmljs2\\\": \"^0.32.0\\\", \"marsdb\\\": \"^0.6.1\\\", \"morgan\\\": \"^1.10.0\\\", \"multer\\\": \"^1.4.5-its.1\\\", \"node-pre-gyp\\\": \"^0.15.0\\\", \"notevil\\\": \"^1.3.3\\\", \"on-finished\\\": \"^2.3.0\\\", \"otplib\\\": \"^12.0.1\\\", \"pdfkit\\\": \"^0.11.0\\\", \"portscanner\\\": \"^2.2.0\\\", \"prom-client\\\": \"^14.1.0\\\", \"pug\\\": \"^3.0.0\\\", \"replace\\\": \"^1.2.0\\\", \"request\\\": \"^2.88.2\\\", \"sanitize-filename\\\": \"^1.6.3\\\", \"sanitize-html\\\": \"^14.2\\\", \"semver\\\": \"^7.3.2\\\", \"sequelize\\\": \"^6.15.1\\\", \"serve-index\\\": \"^1.9.1\\\", \"socket.io\\\": \"^3.1.0\\\", \"sqlite3\\\": \"^5.0.8\\\", \"types/frisby\\\": \"^2.0.10\\\", \"@types/fs-extra\\\": \"^9.0.6\\\", \"@types/glob\\\": \"^7.1.6\\\", \"@types/graceful-fs\\\": \"^4.1.5\\\", \"@types/i18n\\\": \"^0.12.0\\\", \"@types/jasmine\\\": \"^3.9.1\\\", \"@types/jest\\\": \"^26.0.20\\\", \"@types/js-yaml\\\": \"^3.12.6\\\", \"@types/jsonwebtoken\\\": \"^8.5.0\\\", \"@types/jws\\\": \"^3.2.5\\\", \"@types/mocha\\\": \"^8.1.1\\\", \"@types/morgan\\\": \"^1.9.2\\\", \"@types/multer\\\":
```

What type of database is this application using?

This application is using SQLite as its database, as indicated by the presence of "sqlite3": "^5.0.8" in the package file.

How does the application generate HTML?

The application generates HTML using the "pug": "^3.0.0" package. Pug is a high-performance template engine heavily influenced by HamI and implemented with JavaScript for Node.js and browsers.

Filled template

Tokens: 941 Chat

Clear

```
[
{
  "role": "system",
  "content": "You have been provided a package file to analyze. Provide answers to the questions you are given about the application, its libraries in use, and its overall composition.\n\nAnswer the following question: {\\"finale-rest\\": \"^1.1.1\\\", \"fs-extra\\\": \"^9.0.1\\\", \"fuzzball\\\": \"^1.3.0\\\", \"glob\\\": \"^7.1.6\\\", \"graceful-fs\\\": \"^4.2.6\\\", \"grunt\\\": \"^1.2.1\\\", \"grunt-contrib-compress\\\": \"^1.6.0\\\", \"grunt-replace-json\\\": \"^0.1.0\\\", \"hashids\\\": \"^2.2.1\\\", \"hbs\\\": \"^4.0.4\\\", \"helmet\\\": \"^4.0.0\\\", \"html-entities\\\": \"^1.3.1\\\", \"i18n\\\": \"^0.11.1\\\", \"js-yaml\\\": \"^3.14.0\\\", \"jsonwebtoken\\\": \"^0.4.0\\\", \"jssha\\\": \"^3.1.1\\\", \"juicy-chat-bot\\\": \"^0.8.0\\\", \"libxmljs2\\\": \"^0.32.0\\\", \"marsdb\\\": \"^0.6.1\\\", \"morgan\\\": \"^1.10.0\\\", \"multer\\\": \"^1.4.5-its.1\\\", \"node-pre-gyp\\\": \"^0.15.0\\\", \"notevil\\\": \"^1.3.3\\\", \"on-finished\\\": \"^2.3.0\\\", \"otplib\\\": \"^12.0.1\\\", \"pdfkit\\\": \"^0.11.0\\\", \"portscanner\\\": \"^2.2.0\\\", \"prom-client\\\": \"^14.1.0\\\", \"pug\\\": \"^3.0.0\\\", \"replace\\\": \"^1.2.0\\\", \"request\\\": \"^2.88.2\\\", \"sanitize-filename\\\": \"^1.6.3\\\", \"sanitize-html\\\": \"^14.2\\\", \"semver\\\": \"^7.3.2\\\", \"sequelize\\\": \"^6.15.1\\\", \"serve-index\\\": \"^1.9.1\\\", \"socket.io\\\": \"^3.1.0\\\", \"sqlite3\\\": \"^5.0.8\\\", \"types/frisby\\\": \"^2.0.10\\\", \"@types/fs-extra\\\": \"^9.0.6\\\", \"@types/glob\\\": \"^7.1.6\\\", \"@types/graceful-fs\\\": \"^4.1.5\\\", \"@types/i18n\\\": \"^0.12.0\\\", \"@types/jasmine\\\": \"^3.9.1\\\", \"@types/jest\\\": \"^26.0.20\\\", \"@types/js-yaml\\\": \"^3.12.6\\\", \"@types/jsonwebtoken\\\": \"^8.5.0\\\", \"@types/jws\\\": \"^3.2.5\\\", \"@types/mocha\\\": \"^8.1.1\\\", \"@types/morgan\\\": \"^1.9.2\\\", \"@types/multer\\\":
```

What potentially security-impacting or sensitive libraries does this application use?

The application uses several libraries that can have security implications or handle sensitive data:
"jsonwebtoken": "0.4.0" - This library is used to generate JSON Web Tokens, which are often used for authentication and transmitting sensitive information.
"helmet": "4.0.0" - Helmet helps secure Express apps by setting various HTTP headers. It's not a silver bullet, but it can help!
"morgan": "1.10.0" - Morgan is an HTTP request logger middleware for Node.js. It simplifies the process of logging requests to your application.
"multer": "1.4.5-its.1" - Multer is a node.js middleware for handling multipart/form-data, which is primarily used for uploading files.
"otplib": "12.0.1" - It is used for One-Time Passwords (OTP), a common method for two-factor authentication.
"sequelize": "6.15.1" - Sequelize is a promise-based Node.js ORM for Postgres, MySQL, MariaDB, SQLite and Microsoft SQL Server. It features solid transaction support, relations, read replication and more.
"socket.io": "3.1.0"

Baseplate Wrap-up

Good for:

- Testing
- Proof of concept
- Creating a chat agent quickly
- Learning!

Less good for:

- Intense customization
- More than chat, though not impossible by any stretch



Baseplate

Customization using Langchain

Langchain “Chain”

- Pinecone vector store
- OpenAI embeddings
- Prompts (System, Human, Chat)

```
# Convert to an OpenAI Embedding model
pinecone_vector_store = Pinecone.from_existing_index(index_name=pinecone_index,embeddir
print("FINISHED CREATING EMBEDDING MODEL")

# Create system prompt template text
template = '''
Answer questions related to software security.
Supply all answers in a Markdown format suitable for placing inside GitHub issue commer

If you don't know the answer, just say that "I do not yet know the answer to your quest
please ask your security team", don't try to make up an answer.

{summaries}
'''

# Convert text to system prompt
system_message_prompt = SystemMessagePromptTemplate.from_template(template)

# Take the value of question and place it within the human prompt message
human_template = "{question}"
human_message_prompt = HumanMessagePromptTemplate.from_template(human_template)

# Combine our system and human prompts to form our "Chat Message Prompt"
prompt = ChatPromptTemplate.from_messages([system_message_prompt, human_message_prompt])

# Create arguments that will be pushed into our chain
chain_type_kwargs = {"prompt": prompt}
```

Langchain “Chain”

- OpenAI LLM
- Contextual Compression
- Retriever

```
# Choose our openAI chat model and assign a temperature.
llm = ChatOpenAI(model_name="gpt-3.5-turbo", temperature=0.7, max_tokens=256)

# We need to extract and compress the relevant documents so we're giving the
# OpenAI llm as an option here
compressor = LLMChainExtractor.from_llm(llm)

# This has our compressor and vector db options. It is pulling only the relevant text from
# document retrieved based on a user's query rather than the entire document contents.
compression_retriever = ContextualCompressionRetriever(base_compressor=compressor, base_retriever=retriever)

# This performs retrieval of our answers based on our data and context
# and chains together our LLM, our compression retriever, and prompt.
chain = RetrievalQAWithSourcesChain.from_chain_type(
    llm=llm,
    chain_type="stuff",
    retriever=compression_retriever,
    return_source_documents=True,
    chain_type_kwargs=chain_type_kwargs,
    verbose=True
)

response = chain(question)
reply = response["answer"]
print(f"Reply: {reply}")
```

```
cktricky@Kens-MacBook-Pro: ~/code/dryrun/CSA
~/code/d/CSA chat_code 74 python chat_code.py "where is my organization\'s secure coding guide located"
```

Final thoughts on building your own AI powered tooling

🧠 Know what you're trying to use it for

🔗 Integrate into existing tooling and processes

📚 Harness documentation, tribal knowledge, open source security info, etc.

💡 Combine results from many sources to represent an accurate picture of risk

Conclusion

Things are changing... they must

- 💡 Too many technology choices to continue testing in isolation using a single data point to determine risk
- 💡 The power is in your hands to build your own highly customized tooling
- 💡 Meet developers where they live
- 💡 AI / LLM can be a for multiplier; is NOT a silver bullet

Contact Info

@cktricky

ken@dryrun.security

youtube.com/@AbsoluteAppSec

